11-1988

MulCh: a Multi-layer Channel Router using One, Two, and Three Layer Partitions

Ronald I. Greenberg
Loyola University Chicago, Rgreen@luc.edu

Alex T. Ishii

Alberto L. Sangiovanni-Vincentelli

MulCh: A Multilayer Channel Router Using One, Two, and Three Layer Partitions

Ronald I. Greenberg
Electrical Engineering Department
and Institute for Advanced Computer Studies
University of Maryland
College Park, MD 20742

Alexander T. Ishii
Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, MA 02139

Alberto L. Sangiovanni-Vincentelli
EECS Department
University of California
Berkeley, CA 94720

May 1990 revision (plus a few later updates to reference list)

Abstract

Multilayer routing is an important problem in the physical design of integrated circuits as technology evolves towards several layers of metallization. MulCh is a channel router accepting specification of an arbitrary number of routing layers. Though several other channel routers for three layers of interconnect have been proposed, the only previously reported practical implementation for an arbitrary number of layers was that of Chameleon [4]. Chameleon is based on a strategy of decomposing the multilayer problem into two and three layer problems in which one of the layers is reserved primarily for vertical wire runs and the other layer(s) for horizontal runs. In some situations, however, it is advantageous to consider also layers that allow the routing of entire nets, using both horizontal and vertical wires. Hence, MulCh incorporates layers permitting this more general type of wiring. MulCh can route channels with any number of layers and automatically chooses a good assignment of wiring strategies to the different layers. The algorithms have been devised so that MulCh is expected to always perform at least as well as Chameleon in terms of area occupied by the routing. In test cases, MulCh shows significant improvement over Chameleon in terms of channel width, net length, and number of vias.

1 Introduction

Throughout this paper, we discuss the standard grid-based, channel routing problem. Terminals lie on grid points along two horizontal line segments which delimit the channel. Each

This research was supported in part by the Defense Advanced Research Projects Agency under Contracts N00014–87–K–0825 and N00039–87–C–0182. Ron Greenberg was supported in part by a Fannie and John Hertz Foundation Fellowship, and Alberto Sangiovanni-Vincentelli was supported in part by Semiconductors Research Corporation.

A preliminary version of this paper appeared in the IEEE International Conference on Computer-Aided Design in November 1988.

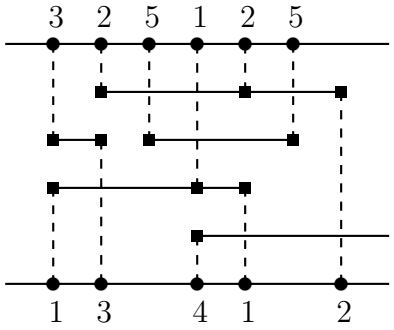

Figure 1: A representative channel routing problem in two layers. The horizontal wires (solid) are in one layer and the vertical (dashed) in the other layer. The vias are represented by squares and the terminals by circles.

terminal is labeled with a net number, and the problem is to connect terminals belonging to the same net, using horizontal and vertical wire segments in a grid of a specified number of layers. (Additionally, some nets may be required to be routed out the left and/or right side of the channel at an unspecified position.) Nets can connect from one layer to another by way of a *contact cut* or *via*; nets cannot intersect one another on the same layer. The primary goal in channel routing is to minimize the channel area. If as is frequently assumed, we cannot extend the channel in the horizontal direction, the objective is to minimize the channel width, the vertical separation between the two lines of terminals. (Variations of the problem, allowing for horizontal shifting of terminals are not considered here.) Secondary goals are to minimize net length and the number of vias. Achieving these secondary goals reduces the time required for signal transmission and increases the probability of fault-free fabrication.

Figure 1 shows a channel routing problem and a solution in two layers, where each layer has been reserved for wires running in one direction (referred to as *reserved layer* or *Manhattan* routing). We refer to each of the vertical grid lines as a column, while the horizontal grid lines are referred to as rows or tracks. We assume here that wires must stay unit distance away from the channel boundaries except when connecting vertically to pins. Thus, the routing shown in Figure 1 achieves the minimum possible channel width when the layers are reserved to individual wiring directions. This optimality follows from the fact that there exist columns which must be crossed by four nets. The convention in this paper is to refer to a routing such as the one in Figure 1 as being of width four, i.e., we count the number of tracks strictly between the lines of terminals.

The lower bound on channel width mentioned in the previous paragraph is one of a few important measures of channel routing difficulty. The bound already alluded to is referred to as the *density* and is generally denoted d. It is the maximum over all columns of the number of nets which must cross the column. A natural variation on this bound will be used in the upcoming discussion of multilayer routing. Another lower bound on channel width in the two-layer Manhattan model is the *flux*, denoted f, which was introduced by Baker, Bhatt, and Leighton [1]. These authors show that any two-layer channel routing problem can be solved in width $2d + O(f)$ (or $d + O(f)$ if all nets have two terminals), and they note that flux appears to be bounded by a small constant for practical problems. This paper will

not make use of flux as a measure of channel routing difficulty; instead it will use an easier to compute measure that has less theoretical justification but is useful in practice. This latter measure is the length of the longest path in the vertical constraint graph (VCG). The VCG is formed by using a node to represent each net and including an edge from net A to net B if a pin for net A appears above a pin for net B in some column of the channel. The length of the longest VCG path is a lower bound on channel width under the Manhattan model with the additional restriction that each net includes only one horizontal wiring segment, i,e., "doglegs" are not allowed. In fact, MulCh does make some use of doglegs, but VCG path length will still be useful as a rough indicator of routing difficulty.

The channel routing problem was first introduced by Hashimoto and Stevens [14] in 1971 and has accumulated an extensive history in the literature. A recent survey has been produced by LaPaugh and Pinter [18]. One of the most important lines of classification for existing channel routing algorithms is the distinction between "provably good" algorithms and "practically good" algorithms. That is, an algorithm that produces excellent results in practice may have a terrible worst-case performance in theory due to pathological instances of the problem unlikely to occur in the real world. On the other hand, an algorithm which has a provably good worst-case performance may virtually always perform worse than a good practical algorithm on "real" problems. The ideal combination of provably and practically excellent performance is unlikely to be attained since most variations of the channel routing problem are generally believed to be NP-complete, and, in fact, rigorous NP-completeness proofs have been provided for various two-layer routing styles [17, 24]. This paper concentrates on a practical, heuristic approach to multilayer channel routing.

Limited results have been presented for multilayer channel routing. The additional layers of interconnect provide more degrees of freedom, but using this freedom effectively is difficult. Chen and Liu [6] proposed an extension to three layers of the algorithm of Yoshimura and Kuh [25]. Bruell and Sun [5] provided a three layer router based on the greedy algorithm of Rivest and Fiduccia [23]. A three layer algorithm of Heyns [15] actually included a departure from the approach of reserving each layer to a single routing direction; such a departure is an important feature of MulCh. Three and four layer routing algorithms have also been provided by Cong, Wong, and Liu [7]. Enbody and Du [10] have proposed a multilayer algorithm which has been implemented and tested in the three and five layer cases. Additional works on multilayer routing have considered more restrictive models of allowable wiring than are considered here or have been geared more towards the goal of theoretical efficacy rather than practical efficacy [2, 3, 13].[1]

Recently, a multilayer channel router, Chameleon [4], was developed to handle any number of interconnection layers. On various sample problems, Chameleon obtains performance generally superior to the preexisting routers. It can handle channel routing problems with cyclic vertical constraints, with different design rules on different layers and with constraints on the contact locations, i.e., stacked vias can be permitted or disallowed.

The basic approach of Chameleon is to divide the original multilayer problem into essentially independent subproblems which are assigned at most three layers each. Chameleon

[1]There are also many other references we have omitted which deal with the "knock-knee" model of routing, in which essentially no overlap of wires on different layers is allowed. For example, see [16].

requires that each layer be reserved primarily for either horizontal wire runs (an H layer) or vertical wire runs (a V layer). The original problem is decomposed by assigning the nets to subproblems, or *groups*, each of which is assigned either one H and one V layer (an HV or VH group) or two H layers and one V layer (an HVH group). Only in the late stages of detailed routing is it possible that wires will digress from this cast. This strategy was inspired by the implicit assumption that the density of the channel is in general much larger than the number of interconnection layers. However, there are situations especially in printed-circuit boards and hybrid circuits where better area utilization can be achieved if sets of nets are routed entirely on one layer.

MulCh is a multilayer channel router that optimizes area utilization by allowing a third type of group, which is assigned a single layer on which wire runs in both the horizontal and vertical directions are permitted (a B layer). This creates obvious complications in that nets assigned to a B layer must not cross if the group is to be routed independently. Use of B layers does, however, give more flexibility and thus may enable a reduction in the area of the route. Possible reductions in total net length and the number of vias are additional benefits.

Like Chameleon, MulCh is composed of two major parts, the partitioner and the detailed router. The partitioner determines the group types used (e.g. HV, HVH, and B) and assigns nets to each of them. The detailed router actually places all the wires in the channel as required to connect pins lying on the same nets. Sections 2 and 3 of this paper describe the partitioner and detailed router as implemented in MulCh. Section 4 provides experimental results, and section 5 contains concluding remarks.

2 Partitioning the Problem

The partitioner has two basic tasks. It must choose the group types (e.g., HVH, HV, and B), and it must assign the nets to the groups. In Chameleon, these two tasks are totally separated; the choice of group types depends only on the number of layers (and for nonuniform technology, the mask information). In MulCh, complete separation of these tasks is not possible since the use of B layers introduces uncontrovertible constraints on net assignment. MulCh seeks a good partition by performing a small number of iterations of the two-part process of first choosing group types and then assigning nets. Thus, the choice of group types can be discussed without delving into the details of net assignment, which will be covered afterwards.

For simplicity, attention is restricted to the case of uniform technology with stacked vias allowed. Other cases could be handled with little additional complication.

2.1 Choosing the Group Types

Chameleon is able to choose group types independently from net assignment, because the best results are generally obtained with the largest possible number of H layers. That is, the partitioning strategy and the detailed router are good enough that the resulting channel

width is usually very close to the density divided by the number of H layers (the lower bound under strict adherence to the H-V wiring model). Thus the strategy of Chameleon is to use as many HVH groups as possible and up to two HV groups.

MulCh follows the same approach once the number of B layers is set, but it is not as easy to determine how many B layers to use. Using more B layers reduces the standard lower bound on channel width (density divided by the number of layers which permit horizontal routing), but an excessive number of B layers may prevent us from coming as close to this lower bound because of the additional constraints on net assignment and detailed routing.[2]

Taking these two competing considerations into account leads naturally to the following strategy. We start with no B layers, assign nets to the groups, and estimate the area required to complete the routing. Then we add the minimum number of B layers necessary to increase the total number of layers which permit horizontal routing, and again assign nets and estimate the area. We repeat this process as long as the area estimate shows improvement[3] and then settle on the number of B layers which yielded the best area estimate. For example, if the number of layers is seven, we start with group types HVH, HV, HV. Then we try HVH, HVH, B, which increases the number of layers permitting horizontal routing from four to five. If the area has improved, we then try HVH, B, B, B, B, since any intermediate number of B layers will not increase the number of layers permitting horizontal routing. Finally, if the area has again improved, we try seven B layers.

In our experiments, we have almost never obtained improved channel width by deviating from the layer assignment strategy just described. If one is willing to perform some extra work in search of reduced net length or via count or a long-shot chance of reduced area, it is reasonable to try also one and two B layers more than the best number obtained under the scheme above. It should be noted that by starting with zero B layers, we replicate the approach of Chameleon, so we expect that our results will be no worse than those of Chameleon as long as we are reasonably good at estimating area once we have assigned the nets to groups. (The area estimate used for B groups is the actual width as determined by performing the complete route. For HV and HVH groups we base the estimate on density, which is fairly accurate once we have paid attention to other concerns during net assignment.) Also, by starting with few B layers and increasing the number, the running time increases only as we obtain more and more improvement upon Chameleon.

2.2 Net Assignment

Once a tentative set of group types has been chosen, nets are assigned to groups one at a time, starting with those nets belonging to cycles in the vertical constraint graph (VCG),

[2]It is interesting to note that if every net has at least one terminal on each of the top and bottom of the channel, then an algorithm of Dagan, Golumbic, and Pinter [8] can be used to determine the number of layers required if *only* B layers are allowed. But if nets with terminals on only one side of the channel are allowed, the problem becomes NP-complete [20, Section V.2.2]. Furthermore, the algorithm of Dagan, Golumbic, and Pinter does not provide any obvious insight on what to do when the number of layers available is not sufficient to use only B layers, nor does it indicate how to assign nets to layers so as to minimize channel width if the number of available layers is more than required to use only B layers.

[3]The actual requirement is that the new result be at least as good rather than strictly better.

and then proceeding through the others in order of first appearance when moving across the channel from left to right. Each net is initially added to all the groups, and a cost function for each group is computed. Then the net is removed from all groups except the best.

Two factors prove to be of key importance in estimating the channel width required to route an HV or HVH group. First, the density of the nets in the group divided by the number of H layers in the group is a lower bound on the channel width. Secondly, and less critically, the length of the longest path in the VCG of the group is a lower bound on channel width in the absence of detailed routing strategies which use doglegs or go beyond strict adherence to the H-V wiring model.

For B layers, the considerations are different. First and foremost we must be sure that a proposed assignment of a net to a B layer does not yield a nonplanar collection of nets in that group. This check is easy, because we can efficiently determine at the beginning of the partitioning process what all the pairs of crossing nets are. Then when an assignment of net A is proposed, we can look at each of the nets crossed by net A and see if any of them have been assigned to the group under consideration.

The net crossings can be found by marching around the four boundaries of the channel in a "circle" and utilizing a stack of nets (with some extra operations beyond the normal stack abstraction). Actually we first march around one more time in a preprocessing step to determine for each net where its first and last occurrence are in the marching order. Then in the main pass around the channel, each time we arrive at a net whose current occurrence is not its last, we push it on the stack. When we arrive at a net whose current occurrence is not its first, we also scan down through the stack until we find the previous occurrence of the current net and cut it out from the stack. Each of the nets we pass on the way crosses the current net, and we can associate the crossing information we glean with each of the relevant nets. This algorithm determines all of the net crossings in time linear in the number of net crossings (multiple crossings of the same nets count) plus the number of positions on the channel boundary.

The above discussion of net crossings has glossed over the ordering of the side terminals, which is not fixed in the original input. There is, however, a readily determined ordering (not necessarily unique) of the side terminals which leaves only those net crossings which must exist regardless of the ordering of the side terminals. Such an ordering is determined and fixed for the partitioner, but this will not be taken to impose any restrictions on the detailed router beyond the restrictions imposed by the essential nature of the partition.

Once we know that a set of nets in a B group is planar, density is an important but imperfect measure of channel width. It is still a lower bound on the width required to route the group, but certain bad arrangements of nets may require much larger channel width. For example, the arrangement shown in Figure 2 has density 2 but requires a channel width of 4.

The length of the longest VCG path is of even less relevance to the width of a planar group. The example in Figure 2 serves also as an illustration of large area with minimal VCG path length. Conversely, it is possible to construct examples with arbitrarily long VCG path length that can be routed in width two. (Performing such a route does require

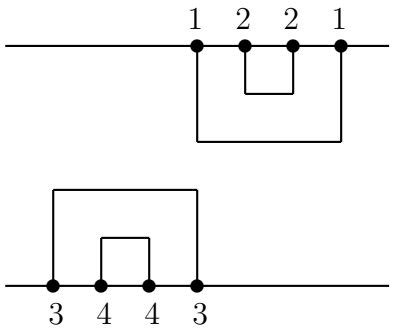

Figure 2: This example illustrates low density and VCG path length but high channel width.

doglegs, but the routines described in Section 3 can compute doglegs for planar routing in polynomial time, in contrast to the NP-complete situation for multilayer groups [24].)

Positively determining the width of a planar group is not straightforward except by essentially performing the complete route. (Section 5 discusses a more efficient method which was not known when MulCh was implemented.) Though the routing can be performed in polynomial time, experience shows that doing this to evaluate every proposed net assignment causes a large increase in running time on large examples but rarely leads to a reduction in area. In practice, density has proven to be a good estimate of the channel width, but blindly holding to this assumption will cause us to obtain worse results than Chameleon on some bad examples. Thus, density is used as the measure of channel width for planar layers during net assignment, but after completing net assignment for a given number of B layers, the B layers are fully routed in order to estimate channel width. This prevents us from making a major gaffe but avoids doing work which is not usually cost-effective.

Another consideration for B groups is that we would like to avoid including nets which will likely stop us from making many future assignments to B groups. MulCh uses a rather crude penalization on assignment of a net to a B group based on the excess over average of the number of nets it crosses.

Chameleon's cost function for evaluating the assignment of a net to a particular partition group involves variously weighted penalty terms based on the notions discussed for two and three layer groups, but MulCh obtains good results using a substantially simpler synthesis of the measures just described. Each HV and HVH group is assigned a cost equal to the maximum of its density and its longest VCG path. Each B group is assigned cost equal to the density plus a penalty of the excess number of nets crossed by the net being assigned. More precisely, for any given group, let d and p denote the density and greatest VCG path length for the group with inclusion of the net currently being assigned. Also, let c denote the number of nets that would be forced to cross the current net if placed together with it in a single-layer (where c is determined in advance independent of any assignment of nets to groups). Then the cost for an HV or HVH group is $\max\{d, p\}$, and the cost for a B group is $d + c - \mu$, where μ is the mean value of c over all nets. The net to be assigned is left in the group which yields the best cost function; ties are broken first on the basis of density, then longest VCG path length, then number of nets in the group.

Several variations on the cost function have been tried. Most lead to improvement on

some example problems and poorer results on other problems. One variant approach actually improves slightly more cases than it worsens but at the expense of some complication in the cost function. In this approach, we include a severe penalty on density exceeding the lower bound on channel width indicated by the number of H and B layers. For HV and HVH groups, we add in the length of the longest VCG path, and this completes the cost function for comparisons among HV and HVH groups. Then in order to make comparisons between these groups and the B groups, we add density into the cost function for each group, and we add the penalty on excess net crossings into the cost for the B groups. Though this variation yields results which are on balance slightly better than those reported in Section 4, there are still several cases which have been routed in less area by some other variation of the program. Thus, the variant results may largely represent chance variations rather than intrinsically superior approaches.

3 Detailed Routing

The detailed router in MulCh is an enhanced version of the detailed router used in Chameleon. The routing of nets in B groups is performed as a preprocessing step, which routes nets using only their assigned layer. Once all nets in B groups have been routed, the HVH and VH routing routines from Chameleon are used to route all remaining nets. Area on a layer assigned to a B group that is not needed to route nets assigned to the group is available for routing other nets.

Routing of nets assigned to B groups is accomplished with a data-structure called a *plane*. A plane is simply a two-dimensional array, with a number of columns equal to the number of columns in the channel, and a number of rows equal to the number of nets in the B group it is used to route. Each entry corresponds to a 2-D grid point on the layer assigned to the group, and thus a row represents a track which may be needed to route the nets in the group. The assumption that the number of tracks needed is less than the number of nets is valid, since the need for more tracks would imply that the nets in the group are not planar.

While not optimal with respect to worst-case running time, the routines for routing B groups are easy to implement and return optimal width routes for the chosen rectilinear wiring model. Worst case running time is not a major concern, since the routing of B groups is fast in comparison to other groups.

The algorithm for planar routing proceeds in a column-by-column sweep starting from the left edge of an "empty" plane. Nets are routed straight across rows of the plane until a change of course is forced by nets which enter or leave the channel at a given column. Nets entering from the top of the channel are routed down the column as far as room permits, while nets entering from the bottom of the channel may need to push other nets up out of the way. When nets are pushed up, the effect is rippled back through previous columns in order to maintain a valid routing. If, on the other hand, a net exits the plane, remaining nets "fall" into the empty space in order to guarantee that the final route will be of minimum width.

One difficulty with the basic plane router is that it places all nets as far towards the bottom of the channel as possible. This tends to generate nets with multiple-doglegs, as well as unnecessary wire runs between the top and bottom of the channel. To alleviate these problems, an optimizing routine is run on the plane after all nets in the B group have been routed. This routine straightens unnecessary doglegs, and moves nets as close as possible to their associated terminals. Figure 3 illustrates some of the operational features of the single layer router.

Nets not assigned to B groups are routed by an amortized version of YACR2 [21] that differs from that used in the original Chameleon only in that nets from other groups cannot initially be placed on a layer assigned to a B group. This restriction is a pragmatic one, based on a desire to avoid unnecessarily long computation times. In practice, the lack of an associated vertical routing layer virtually guarantees that a net not in a particular B group will be unroutable if placed on the group's associated layer. In all other ways, however, unused space on layers assigned to B groups is available for routing other nets.

While no experimental examples were in evidence, it is possible that the introduction of B layers will increase the required channel width, while reducing the number of required vias. A trade-off such as this is most likely to occur when the partitioner estimates that a partition using B layers will have area identical to a partition without B layers. In such cases, the inevitable reduction in available V layers makes the detailed routing more difficult, and thus favors the partition without B layers for minimum area. If area is not the critical parameter, however, the reliability advantages of fewer vias would speak in favor of the use of B layers. In practice, the IC designer should run MulCh with and without the use of B layers enabled, and choose the route that best matches the system design goals.

4 Experimental Results

This section describes experimental results obtained with MulCh as compared to results obtained by Chameleon.[4] Performance comparisons were made using a set of fourteen benchmark examples described below. The most dramatic improvements were seen in the four and seven-layer tests, where the mean reductions in channel width, total net length, and total number of vias were about 8%, 4%, and 20%, respectively.[5] MulCh routed several of the sample channels in width not only less than that of Chameleon, but less than the minimum theoretically possible without B layers.

Table 1 describes the example channels that were used to evaluate the performance of MulCh. Example channels include **3a**, **3b**, and **3c** from [25], the largest channel of the

[4] As stated earlier, MulCh was implemented by enhancing the Chameleon channel routing package. The version of Chameleon used here is actually a sightly revised version of the package described in [4]. Performance of the revised version is broadly equivalent to the earlier package, with isolated examples differing by at most a single row (generally in favor of the revised version). Since one purpose of the presented results is to demonstrate the inherent benefits of B layers, all comparisons are to the version of Chameleon that MulCh is based upon.

[5] These means are computed by first expressing the results of MulCh for each channel as a percentage of the corresponding result for Chameleon. Then a geometric mean is computed and subtracted from 100%.

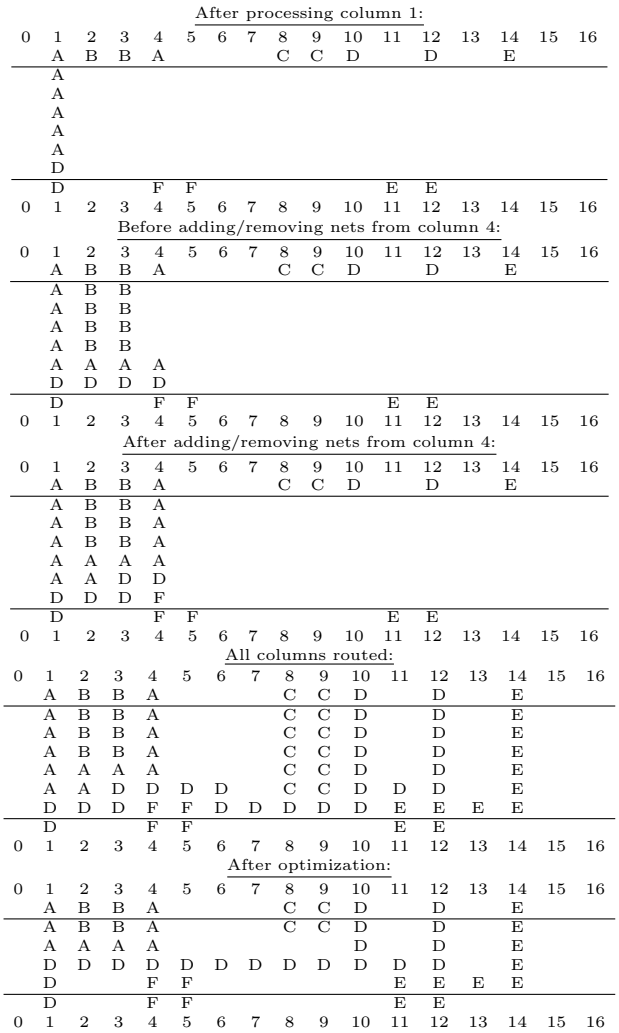

Figure 3: Shown is a plane as it appears at various points during the routing of the indicated channel. The routing of column 4 is shown both before and after the nets for terminals at the column are added or removed. Note the displacement of nets D, A and B that is caused by the entrance of net F. In addition, note how the exit of net F in column 5 allows net D to "drop" into the bottom row of the plane. The final optimization removes redundant rows, unnecessary doglegs (nets A and D) and excess wire runs (net C).

Example	Density	Number of Columns	Number of Nets
3a	15	45	30
3b	17	62	47
3c	18	103	54
chris1	49	432	158
cycle.t	16	134	65
diff	19	157	52
ex1	16	417	235
ex2	15	421	282
ex3	11	421	291
ex4	19	421	270
r1	20	139	77
r2	20	117	77
r3	16	123	78
r4	15	150	74

Table 1: Benchmark channels

CMOS implementation of the SOAR microprocessor designed at Berkeley (**chris1**) [19], the contrived cyclic-VCG example (**cycle.t**) from [4], the well-known Deutsch's difficult example [9] (**diff**), and four random problems (**r1**, **r2**, **r3**, and **r4**) generated by Rivest's program [22]. The examples **ex1** through **ex4** are difficult channels from an industrial standard cell chip provided by C. P. Hsu.

Table 2 lists the performance of MulCh and its improvements over Chameleon. The numbers enclosed in parentheses next to the channel width indicate the number of B layers used on each example. The columns labeled $\Delta\mathbf{R}$, $\Delta\mathbf{NL}$, and $\Delta\mathbf{V}$ give the differences between the number of rows, total net length, and total number of vias needed by MulCh and Chameleon to route a particular example. For convenience, entries under $\Delta\mathbf{NL}$ and $\Delta\mathbf{V}$ are listed as percentage reductions in Chameleon's results which were achieved by MulCh. Though all fourteen of the examples were tested on two through seven layers, no B layers were used by MulCh in the cases omitted from the table. In these cases results obtained with MulCh are comparable, but not necessarily identical, to the results obtained by Chameleon. The previously mentioned simplifications to the partitioner cost function are responsible for the discrepancies.

The most interesting results are for four and seven layers, where the ability to easily increase the number of layers available for horizontal routing encourages the use of B layers. In the four-layer scenario, especially, MulCh often succeeded in simultaneously reducing the number of rows and vias, as well as total net length. Reductions in total net length and number of vias were also evident in cases where the number of rows was not reduced. The four-layer table is particularly relevant, since it represents technology which is currently becoming available. In the table for seven layers, fewer row number reductions appear than in the four-layer table; savings came mostly as reductions in total-net-length and the number of vias.

MulCh found only one opportunity for improvement when five and six layers were em-

Ex.	Width		Net Length		Vias	
4 Lyr.	New	ΔR	New	ΔNL	New	ΔV
3a	6(1)	2*	744	9.9%	53	24.3%
3b	7(1)	2*	1227	6.5%	85	17.5%
3c	8(1)	1*	1903	5.6%	126	15.4%
chris1	23(1)	2*	17128	1.3%	346	16.0%
cycle.t	8(1)	1	2676	1.1%	217	4.0%
diff	9(1)	1	3129	1.2%	227	0.9%
ex1	8(1)	0	5342	3.1%	348	22.7%
ex2	7(1)	1*	5278	7.9%	349	31.7%
ex3	5(1)	1*	4363	7.5%	317	27.8%
ex4	9(1)	1*	5273	7.6%	316	29.0%
r1	10(1)	1	3285	4.0%	170	20.2%
r2	10(1)	0	2579	2.3%	124	23.5%
r3	8(1)	0	2248	2.6%	151	19.3%
r4	8(1)	0	2803	2.7%	199	18.4%
5 Lyr.	New	ΔR	New	ΔNL	New	ΔV
ex3	4(2)	0	4013	6.4%	192	56.3%
7 Lyr.	New	ΔR	New	ΔNL	New	ΔV
3a	4(1)	0	691	2.8%	57	16.2%
3b	4(1)	1*	1071	6.1%	83	19.4%
3c	5(1)	0	1670	1.8%	117	21.5%
chris1	12(1)	1*	15288	1.7%	345	16.3%
cycle.t	4(1)†	0	2203	3.8%	198	16.1%
diff	5(1)	0	2657	-0.9%	207	8.9%
ex1	4(1)†	0	4452	3.7%	321	28.3%
ex2	4(1)	0	4474	4.1%	337	32.2%
ex3	3(0)†	1	4030	5.0%	439	-0.5%
ex4	5(1)	0	4356	2.9%	338	24.7%
r1	5(1)	1	2858	3.3%	170	17.9%
r2	5(1)	0	2206	3.7%	128	21.0%
r3	4(1)†	1	1861	5.9%	150	20.2%
r4	4(1)	0	2431	3.8%	197	18.6%

Table 2: Results with MulCh

ployed. For five and six layers, the number of layers available for horizontal routing does not increase with the addition of only a single B layer, and consequently the partitioner requires many coplanar nets to be successful.

Table 2 contains several entries worth noting. Of particular interest are the $\Delta\mathbf{R}$ entries marked with an asterisk, which indicate improvements over the optimal solutions obtainable without the use of B layers. Also of interest is example **ex3** for five layers, where a 56.3% reduction in vias was realized.

In only a few of the cases shown in the table (as indicated by a dagger next to the width), did MulCh achieve channel widths meeting the naive lower bound of density divided by the number of H and B layers. This is to be expected, since the planarity requirement greatly restricts the set of nets that can be placed in a B group.

While MulCh's heuristic-search nature makes a complete analysis of running time impossible, it should be noted that all examples required less than eight minutes of time on a Microvax III computer. Additional code optimizations should result in further running time reductions.

5 Conclusion

The main conclusion to be drawn from our rsearch is that existing multilayer channel routers can benefit from the incorporation of layers which are not dedicated to a single routing direction. Application of this approach to Chameleon has yielded significant improvements on example problems in terms of channel width, total net length, and total number of vias.

There are several further areas of research, including the use of more efficient or more powerful algorithms for some of the subtasks performed by MulCh. As mentioned earlier, it is actually possible to determine minimum required channel width for a single-layer problem in a more efficient fashion than performing the complete routing. In fact, linear time suffices to determine minimum channel width [12, 11]. Thus, it becomes feasible to determine the true minimum width of B layers each time that a net assignment is considered, instead of relying on the density estimate until the end of net assignment. This change could lead to a better quality of net partitions. Improvements in the running time of MulCh might be obtained by also incorporating incremental algorithms into the net assignment process. For example, recomputing the density of a partition when a new net is added requires only logarithmic time rather than the naive linear time [11].

An additional useful area of investigation would be to compare the channel router presented in this paper to the routing algorithm of Berger, Brady, Brown, and Leighton [2], which has a "provably good" worst-case performance. Their algorithm applied to an L-layer problem of density d will produce a routing of width $\frac{d}{L-2} + O(\sqrt{d/L} + 1)$, which could be quite attractive depending on the exact size of the additive term and the ability of the algorithm to do better than the worst case on practical problems. It would be desirable to create a practically oriented implementation of this algorithm and to ascertain its performance on example problems.

Acknowledgements

We wish to thank all those who participated in the development of Chameleon, which provided the core for our own code. Special thanks to Fabio Romeo of Berkeley for helpful advice and technical assistance.

References

[1] B. S. Baker, S. N. Bhatt, and F. T. Leighton. An approximation algorithm for Manhattan routing. In F. P. Preparata, editor, *VLSI Theory*, volume 2 of *Advances in Computing Research*, pages 205–229. JAI Press, 1984.

[2] B. Berger, M. Brady, D. Brown, and T. Leighton. Nearly optimal algorithms and bounds for multilayer channel routing, 1988. Manuscript. Later version in *JACM* 42(2), pp. 500–542, Mar 1995.

[3] M. L. Brady and D. J. Brown. Optimal multilayer channel routing with overlap. In C. E. Leiserson, editor, *Advanced Research in VLSI: Proceedings of the Fourth MIT Conference*, pages 281–296. MIT Press, 1986. Later version in *Algorithmica*, 1991.

[4] D. Braun, J. L. Burns, F. Romeo, A. Sangiovanni-Vincentelli, K. Mayaram, S. Devadas, and H.-K. T. Ma. Techniques for multi-layer channel routing. *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, 7(6):698–712, June 1988.

[5] P. Bruell and P. Sun. A "greedy" three layer channel router. In *Proceedings of the IEEE International Conference on Computer-Aided Design (ICCAD-85)*, pages 298–300. IEEE Computer Society Press, 1985.

[6] Y. K. Chen and M. L. Liu. Three-layer channel routing. *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, CAD-3(2):156–163, Apr. 1984.

[7] J. Cong, D. F. Wong, and C. L. Liu. A new approach to three- or four-layer channel routing. *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, 7(10):1094–1104, Oct. 1988.

[8] I. Dagan, M. C. Golumbic, and R. Y. Pinter. Trapezoid graphs and their coloring. *Discrete Applied Mathematics*, 21:35–46, 1988.

[9] D. Deutsch. A 'dogleg' channel router. In *Proceedings of the 13th ACM/IEEE Design Automation Conference*, pages 425–433. IEEE Computer Society Press, 1976.

[10] R. J. Enbody and H. C. Du. Near-optimal n-layer channel routing. In *Proceedings of the 23rd ACM/IEEE Design Automation Conference*, pages 708–714. IEEE Computer Society Press, 1986.

[11] R. I. Greenberg. *Efficient Interconnection Schemes for VLSI and Parallel Computation.* PhD thesis, Department of Electrical Engineering & Computer Science, Massachusetts Institute of Technology, Aug. 1989. MIT/LCS/TR-456.

[12] R. I. Greenberg and F. M. Maley. Finding minimum width for single-layer channel routing in linear time. Technical Report UMIACS-TR-90-4, University of Maryland Institute for Advanced Computer Studies, Jan. 1990. Later version in *Information Processing Letters* 43(4), pp. 201–205, Sep 1992.

[13] S. E. Hambrusch. Channel routing algorithms for overlap models. *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, CAD-4(1):23–30, Jan. 1985.

[14] A. Hashimoto and J. Stevens. Wire routing by optimizing channel assignment within large apertures. In *Proceedings of the 8th ACM/IEEE Design Automation Conference*, pages 155–169. IEEE Computer Society Press, 1971.

[15] W. Heyns. The 1-2-3 routing algorithm or the single channel 2-step router on 3 interconnection layers. In *Proceedings of the 19th ACM/IEEE Design Automation Conference*, pages 113–120. IEEE Computer Society Press, 1982.

[16] R. Kuchem, D. Wagner, and F. Wagner. Area-optimal three-layer channel routing. In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science*, pages 506–511. IEEE Computer Society Press, 1989.

[17] A. S. LaPaugh. *Algorithms for Integrated Circuit Layout: An Analytic Approach.* PhD thesis, Department of Electrical Engineering & Computer Science, Massachusetts Institute of Technology, Nov. 1980. MIT/LCS/TR-248.

[18] A. S. LaPaugh and R. Y. Pinter. Channel routing for integrated circuits. *Annual Review of Computer Science*, 4:307–363, 1989.

[19] C. Marino. Smalltalk on a RISC - CMOS implementation. MS report, Department of EECS, University of California, Berkeley, 1985.

[20] R. Y. Pinter. *The Impact of Layer Assignment Methods on Layout Algorithms for Integrated Circuits.* PhD thesis, Department of Electrical Engineering & Computer Science, Massachusetts Institute of Technology, Aug. 1982. MIT/LCS/TR-291.

[21] J. Reed, A. Sangiovanni-Vincentelli, and M. Santomauro. A new symbolic channel router: YACR2. *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, CAD-4(3):208–219, July 1985.

[22] R. L. Rivest. "Benchmark" channel-routing problems. Manuscript, 1982.

[23] R. L. Rivest and C. M. Fiduccia. A "greedy" channel router. In *Proceedings of the 19th ACM/IEEE Design Automation Conference*, pages 418–424. IEEE Computer Society Press, 1982.

[24] T. G. Szymanski. Dogleg channel routing is NP-complete. *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, CAD-4(1):31–41, Jan. 1985.

[25] T. Yoshimura and E. S. Kuh. Efficient algorithms for channel routing. *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, CAD-1(1):25–35, Jan. 1982.